Interpretation der Zeitmessung

```
-------------------------Ab hier beginnt die List Liste-------------------------
3ms | addEnd
57ms | addFirst
633ms | DeleteHalf
3ms | summeIterator | Ergebnis = 1972224
3ms | summeIndex | Ergebnis = 1972224
387ms | listeIterator
385ms | listeIndex
-------------------------Ab hier endet die List Liste-------------------------
-------------------------Ab hier beginnt die Collection Liste-------------------------
1ms | addEnd
47ms | addFirst
450ms | DeleteHalf
1ms | summeIterator | Ergebnis = 1966678
1ms | summeIndex | Ergebnis = 1966678
196ms | listeIterator
191ms | listeIndex
-------------------------Ab hier endet die Collection Liste-------------------------
```

AddEnd/AddFirst

Bei AddEnd – Collection/List: Diese Methode funktioniert bei beiden Collections sehr schnell, da die neuen Elemente nur an die bestehende Liste hinten darangefügt werden müssen.

Bei AddFirst – Collection/List: Diese Methode dauert deutlich länger, da die neuen Elemente an den ersten Platz der Liste hinzugefügt werden müssen und dadurch alle Elemente um eins nach hinten verschoben werden müssen, was bei 20000 Elementen recht zeitaufwändig ist.

DeleteHalf

Wie vorher dauert das Löschen der hälfte der Elemente des Arrays bei beiden Listen gleichlang. Hierbei erkennt man, dass diese Methode viel mehr Rechenleistung/Zeit in Anspruch nimmt als AddEnd/AddFirst. Das kommt zu Stande, da ja die ganze Liste durchgegangen werden muss und zusätzlich jede zweite Stelle gelöscht werden soll. UND nach einer Löschung müssen die anderen Zahlen wieder an die richtige Position verschoben werden.

SummeIterator/SummeIndex

Wie vorher dauert die Berechnung der Summe der Elemente des Arrays bei den beiden Listen gleichlang. Diese Methode ist nicht sehr Leistung/Zeitaufwändig.

Da die beiden Methoden einem simplen Mechanismus folgen: Es wird jedes Element der List durchgegangen und dann in eine Summenvariable gespeichert und aufaddiert.

Die Methoden an sich unterscheiden sich fast gar nicht. Die Iterator Methode hat eine eigene Schleife, die auf dem Prinzip der Index-Schleife basiert.

ListeIterator/ListeIndex

Die beiden Methoden nehmen viel Rechenleistung/Zeit in Anspruch. Da jedes einzelne Element der Liste durchgegangen, in eine String Variable gespeichert/aufaddiert und ausgegeben werden muss kommt diese Statistik zusammen.

Hier ein paar mehr Testversuche:

```
-------------------------Ab hier beginnt die List Liste-------------------------
3ms | addEnd
52ms | addFirst
611ms | DeleteHalf
2ms | summeIterator | Ergebnis = 1987230
3ms | summeIndex | Ergebnis = 1987230
386ms | listeIterator
353ms | listeIndex
-------------------------Ab hier endet die List Liste-------------------------
-------------------------Ab hier beginnt die Collection Liste-------------------------
1ms | addEnd
48ms | addFirst
457ms | DeleteHalf
2ms | summeIterator | Ergebnis = 1985827
2ms | summeIndex | Ergebnis = 1985827
192ms | listeIterator
198ms | listeIndex
-------------------------Ab hier endet die Collection Liste-------------------------
```

```
-------------------------Ab hier beginnt die List Liste-------------------------
4ms | addEnd
51ms | addFirst
592ms | DeleteHalf
2ms | summeIterator | Ergebnis = 1980227
2ms | summeIndex | Ergebnis = 1980227
392ms | listeIterator
392ms | listeIndex
-------------------------Ab hier endet die List Liste-------------------------
-------------------------Ab hier beginnt die Collection Liste-------------------------
1ms | addEnd
47ms | addFirst
455ms | DeleteHalf
2ms | summeIterator | Ergebnis = 1968725
2ms | summeIndex | Ergebnis = 1968725
198ms | listeIterator
198ms | listeIndex
-------------------------Ab hier endet die Collection Liste-------------------------
```